Higher Order Force Gradient Symplectic Algorithms

Siu A. Chin and Donald W. Kidwell*

Center for Theoretical Physics, Department of Physics, Texas A&M University, College Station, TX 77843

(July 4, 2000)

Abstract

We show that a recently discovered fourth order symplectic algorithm, which requires one evaluation of force gradient in addition to three evaluations of the force, when iterated to higher order, yielded algorithms that are far superior to similarly iterated higher order algorithms based on the standard Forest-Ruth algorithm. We gauge the accuracy of each algorithm by comparing the step-size independent error functions associated with energy conservation and the rotation of the Laplace-Runge-Lenz vector when solving a highly eccentric Kepler problem. For orders 6, 8, 10 and 12, the new algorithms are approximately a factor of 10^3, 10^4, 10^4 and 10^5 better.

Typeset using REVTEX

*Present address: Candescent Technologies, 6580 Via Del Oro, San Jose, CA 95119

I. INTRODUCTION

Symplectic algorithms [1,2] for solving classical dynamical problems exactly conserve all Poincaré invariants. For periodic orbits, the errors in energy conservation are bounded and periodic. This is in sharp contrast to Runge-Kutta type algorithms, whose energy error increases linearly with integration time, even for periodic orbits [3,4]. Thus, symplectic algorithms are ideal for long time integration of equations of motion in problems of astrophysical interest [5]. For long time integrations, higher order algorithms are desirable because they permit the use of larger time steps. Symplectic algorithms are also advantageous in that higher order algorithms can be systematically generated from any low, even order algorithm [6–8]. In this work, we will show that higher order algorithms generated by a fourth order force gradient symplectic algorithm [9], have energy errors that are several orders of magnitude smaller than existing symplectic algorithms of the same order. For completeness, we will briefly summarize the operator derivation of symplectic algorithms and their higher order construction in Section II. While the materials in this section are not new, we believe that we have restated Creutz' and Gockach's [6] triplet construction of higher order algorithms in its most transparent setting. In sections III and IV we recall force gradient algorithms and discuss two distinct ways of gauging the errors of an algorithm when solving the Kepler problem. We present our results and conclusions in Sections V and VI.

II. OPERATOR FACTORIZATION AND HIGHER ORDER CONSTRUCTIONS

After a tortuous start [10,11], symplectic algorithms can be derived most simply on the basis of operator decomposition or factorization. For any dynamical variable $W(q_i, p_i)$, its time evolution is given by the Poisson bracket,

$$\frac{d}{dt}W(q_i,p_i) = \{W,H\} \equiv \sum_i \Big(\frac{\partial W}{\partial q_i}\frac{\partial H}{\partial p_i} - \frac{\partial W}{\partial p_i}\frac{\partial H}{\partial q_i}\Big). \tag{1}$$

If the Hamiltonian is of the form,

$$H(p,q) = \frac{1}{2}\sum_i p_i^2 + V(\{q_i\}), \tag{2}$$

the evolution equation (1) can be written as an operator equation

$$\frac{dW}{dt} = \sum_i \Big(p_i \frac{\partial}{\partial q_i} + F_i \frac{\partial}{\partial p_i}\Big) W, \tag{3}$$

with formal solution

$$W(t) = \mathrm{e}^{t(T+V)} W(0), \tag{4}$$

where T and V are first order differential operators defined by

$$T \equiv \sum_i p_i \frac{\partial}{\partial q_i}, \qquad V \equiv \sum_i F_i \frac{\partial}{\partial p_i}. \tag{5}$$

Their exponentiations, $\mathrm{e}^{\epsilon T}$ and $\mathrm{e}^{\epsilon V}$, are displacement operators which displace q_i and p_i forward in time via

$$q_i \to q_i + \epsilon p_i \qquad \text{and} \qquad p_i \to p_i + \epsilon F_i. \tag{6}$$

Thus, if $\mathrm{e}^{\epsilon(T+V)}$ can be factorized into products of the displacement operators $\mathrm{e}^{\epsilon T}$ and $\mathrm{e}^{\epsilon V}$, each such factorization gives rise to an algorithm for evolving the system forward in time. For example, the second order factorization

$$\mathcal{T}_2(\epsilon) \equiv \mathrm{e}^{\frac{1}{2}\epsilon T}\mathrm{e}^{\epsilon V}\mathrm{e}^{\frac{1}{2}\epsilon T} = \exp[\epsilon(T+V) + \epsilon^3 C + +O(\epsilon^5)\cdots], \tag{7}$$

corresponds to the second order algorithm

$$\begin{aligned} \mathbf{q}_1 &= \mathbf{q}_0 + \frac{1}{2}\epsilon \mathbf{p}_0, \\ \mathbf{p}_1 &= \mathbf{p}_0 + \epsilon \mathbf{F}(\mathbf{q}_1), \\ \mathbf{q}_2 &= \mathbf{q}_1 + \frac{1}{2}\epsilon \mathbf{p}_1, \end{aligned} \tag{8}$$

where $\mathbf{q}_0$, $\mathbf{p}_0$ and $\mathbf{q}_2$, $\mathbf{p}_1$ are the initial and final states of the algorithm respectively. This second order symplectic algorithm only requires one evaluation of the force.

The bilaterally symmetric form of $\mathcal{T}_2(\epsilon)$ automatically guarantees that it is *time-reversible*,

$$\mathcal{T}_2(-\epsilon)\mathcal{T}_2(\epsilon) = 1, \tag{9}$$

and implies that $\log(\mathcal{T}_2)$ can only be an odd function of ϵ, as indicated in (7). The explicit form of the operator C is not needed for our present discussion.

Consider now the the symmetric triple product

$$\mathcal{T}_2(\delta)\mathcal{T}_2(-s\delta)\mathcal{T}_2(\delta) = \exp[(2-s)\delta(T+V) + (2-s^3)\delta^3 C + O(\delta^5) + \cdots]. \tag{10}$$

This algorithm evolves the system forward for time δ, backward for time $s\delta$ and forward again for time δ. Since it is manifestly time-reversible, its error terms must be odd powers of δ only. Morever, its leading first and third order terms can only be the sum of the first and third order terms of each constituent algorithm as indicated. This is because non-additive terms must come from commutators of operators and the lowest order non-vanishing commutator has to have two first order terms and one third order term, which is fifth order. The form of (10) naturally suggests that the third order error term can be made to vanish by choosing

$$s = 2^{\,1/3}. \tag{11}$$

Thus if we now rescale δ back to the standard step size by setting $\epsilon = (2-s)\delta$, the resulting triplet product would be correct to 4th order,

$$\mathcal{T}_4 \equiv \mathcal{T}_2\Big(\frac{\epsilon}{2-s}\Big)\mathcal{T}_2\Big(\frac{-s\,\epsilon}{2-s}\Big)\mathcal{T}_2\Big(\frac{\epsilon}{2-s}\Big) = \exp[\,\epsilon(T+V) + O(\epsilon^5) + \cdots]. \tag{12}$$

Expanding out the $\mathcal{T}_2$'s gives the explicit form:

$$\mathcal{T}_4 \equiv \mathrm{e}^{a_1\epsilon T}\mathrm{e}^{b_1\epsilon V}\mathrm{e}^{a_2\epsilon T}\mathrm{e}^{b_2\epsilon V}\mathrm{e}^{a_2\epsilon T}\mathrm{e}^{b_1\epsilon V}\mathrm{e}^{a_1\epsilon T} \tag{13}$$

where, by inspection

$$a_1 = \frac{1}{2}\frac{1}{2-s}, \quad a_2 = -\frac{1}{2}\frac{s-1}{2-s}, \quad b_1 = \frac{1}{2-s}, \quad \text{and} \quad b_2 = -\frac{s}{2-s}. \tag{14}$$

This fourth order symplectic algorithm was apparently obtained by E. Forest in 1987. However, its original derivation was very complicated and was not published with Ruth [11] until 1990. During this period many groups, including Campostrini and Rossi [12] in 1990,

Candy and Rozmous [13] in 1991, independently published the same algorithm. Our discussion followed the earliest published derivation of this algorithm by Creutz and Gocksch [6] in 1989. After they were informed of this algorithm by Campostrini, they provided the triplet construction and generalized it to higher order. The triplet construction was also independently published by Suzuki [7] and Yoshida [8] in 1990.

Higher order algorithms can be obtained by repeating this construction. Starting with any nth order symmetric algorithm,

$$\mathcal{T}_n(\epsilon) = \exp[\,\epsilon(T+V) + \epsilon^{(n+1)}D + \cdots], \tag{15}$$

the triplet product

$$\mathcal{T}_n(\delta)\mathcal{T}_n(-s\delta)\mathcal{T}_n(\delta) = \exp[(2-s)\delta(T+V) + (2-s^{n+1})\delta^{n+1}D + O(\delta^{n+3}) + \cdots], \tag{16}$$

will be of order $(n+2)$ if we choose

$$s = 2^{\,1/(n+1)} \tag{17}$$

and renormalize $\delta = \epsilon/(2-s)$ as before.

III. FORCE GRADIENT ALGORITHMS

The method of operator factorization can be applied to many different classes of evolution equations. However, the triplet concatenations with a negative time step are a special construction with more limited applicability. For example, one cannot use it to derive similar Diffusion Monte Carlo or finite temperature path integral algorithms, because one cannot simulate diffusion backward in time nor sample configurations with negative temperatures. The triplet construction is a special example of Suzuki's [14] general proof that, beyond second order, it is impossible to factorize $\mathrm{e}^{\epsilon(T+V)}$ only into products of $\mathrm{e}^{\epsilon T}$'s and $\mathrm{e}^{\epsilon V}$'s without introducing negative time steps. For symplectic algorithms this means that one can never develop a purely positive time step fourth order algorithm by evaluating only the force. For many years the Forest-Ruth (FR) algorithm was the only known fourth

order symplectic algorithm. Recently, a deeper understanding of the operator factorization process has yielded three new symplectic algorithms [9] all with purely positive time steps. These new algorithms circumvented Suzuki's no-go theorem by evaluating the force and its gradient. This corresponds to factorizing $\mathrm{e}^{\epsilon(T+V)}$ in terms of operators T, V, and the commutator $[V,[T,V]]$. The latter corresponds to

$$[V,[T,V]] = 2F_j\frac{\partial F_i}{\partial q_j}\frac{\partial}{\partial p_i} = \nabla_i|\mathbf{F}|^2\frac{\partial}{\partial p_i}, \tag{18}$$

which is the gradient of the squared magnitude of the force. Of the three algorithms derived by Chin [9], algorithm C is particularly outstanding and corresponds to the factorization

$$\mathrm{e}^{\epsilon(T+V)} = \mathrm{e}^{\epsilon\frac{1}{6}T}\mathrm{e}^{\epsilon\frac{3}{8}V}\mathrm{e}^{\epsilon\frac{1}{3}T}\mathrm{e}^{\epsilon\frac{1}{4}\widetilde{V}}\mathrm{e}^{\epsilon\frac{1}{3}T}\mathrm{e}^{\epsilon\frac{3}{8}V}\mathrm{e}^{\epsilon\frac{1}{6}T} + O(\epsilon^5). \tag{19}$$

where

$$\widetilde{V} = V + \frac{1}{48}\epsilon^2[V,[T,V]]. \tag{20}$$

The algorithm itself can be read off directly as

$$\begin{aligned}
\mathbf{q}_1 &= \mathbf{q}_0 + \frac{1}{6}\epsilon\mathbf{p}_0,\\
\mathbf{p}_1 &= \mathbf{p}_0 + \frac{3}{8}\epsilon\mathbf{F}(\mathbf{q}_1)\\
\mathbf{q}_2 &= \mathbf{q}_1 + \frac{1}{3}\epsilon\mathbf{p}_1\\
\mathbf{p}_2 &= \mathbf{p}_1 + \frac{1}{4}\epsilon\Big[\mathbf{F}(\mathbf{q}_2) + \frac{1}{48}\epsilon^2\nabla|\mathbf{F}(\mathbf{q}_2)|^2\Big]\\
\mathbf{q}_3 &= \mathbf{q}_2 + \frac{1}{3}\epsilon\mathbf{p}_2\\
\mathbf{p}_3 &= \mathbf{p}_2 + \frac{3}{8}\epsilon\mathbf{F}(\mathbf{q}_3)\\
\mathbf{q}_4 &= \mathbf{q}_3 + \frac{1}{6}\epsilon\mathbf{p}_3.
\end{aligned} \tag{21}$$

In [9] it was shown that the maximum energy error for this algorithm, when used to solve Kepler's problem, is smaller than that of the FR algorithm by a factor of 80. At the moment there is no general method for constructing higher order algorithms with only positive time steps. It is not even known whether a positive time step 6th order algorithm exists. Thus,

beyond 4th order the triplet contruction is still the only systematic way of generating higher order algorithms. In this work we show that intrinsic error functions associated with higher order algorithms generated from Chin's algorithm C are far smaller than those generated from the FR algorithm.

IV. THE ENERGY AND THE LRL VECTOR

We gauge the numerical effectiveness of each algorithm by solving the two dimensional Kepler problem

$$\frac{d^2\mathbf{q}}{dt^2} = -\frac{\mathbf{q}}{q^3}, \tag{22}$$

with initial conditions $\mathbf{q}_0 = (10, 0)$ and $\mathbf{p}_0 = (0, 1/10)$. The resulting highly eccentric (e=0.9) orbit provides a non-trivial testing ground for trajectory integration.

A symmetric nth order symplectic algorithm evolves this system forward in time with Hamiltonian

$$H(\mathbf{p}, \mathbf{q}) = H_0(\mathbf{p}, \mathbf{q}) + \epsilon^n H_n(\mathbf{p}, \mathbf{q}) + O(\epsilon^{n+2}), \tag{23}$$

which deviates from the exact Hamiltonian $H_0(\mathbf{p}, \mathbf{q}) = \frac{1}{2}\mathbf{p}^2 - 1/|\mathbf{q}|$ by an error term $\epsilon^n H_n(\mathbf{p}, \mathbf{q})$ as indicated. To gauge the intrinsic merit of each algorithm we compare their step-size independent error coefficient $H_n(\mathbf{p}, \mathbf{q})$. This can be extracted numerically as follows. Let's start the system with total energy $E_0 = H_0(\mathbf{p}(0), \mathbf{q}(0))$. Since the Hamiltonian (23) is conserved by the algorithm, we have

$$E_0 = H_0(\mathbf{p}(t), \mathbf{q}(t)) + \epsilon^n H_n(\mathbf{p}(t), \mathbf{q}(t)) + O(\epsilon^{n+2}) \tag{24}$$

Denoting $E(t) \equiv H_0(\mathbf{p}(t), \mathbf{q}(t))$ and $H_n(t) = H_n(\mathbf{p}(t), \mathbf{q}(t))$, we therefore have

$$H_n(t) = -\lim_{\epsilon \to 0} \frac{1}{\epsilon^n}\Big[E(t) - E_0\Big]. \tag{25}$$

Energy conservation does not directly measure how well the orbit is determined. When the time step is not too small, a very noticeable error is that the orbit precesses. One can,

but it is tedious, directly monitor this orbital precession [4]. It is more expedient to follow the rotation of the Laplace-Runge-Lenz (LRL) vector:

$$\mathbf{A} = \mathbf{p} \times \mathbf{L} - \hat{\mathbf{r}}. \tag{26}$$

When the orbit is exact the LRL vector is constant, pointing along the semi-major axis of the orbit. When the orbit precesses the LRL vector rotates correspondingly.

For an nth order algorithm,

$$\frac{d\mathbf{A}}{dt} = \epsilon^n \sum_i \Big(\frac{\partial \mathbf{A}}{\partial q_i} \frac{\partial H_n}{\partial p_i} - \frac{\partial \mathbf{A}}{\partial p_i} \frac{\partial H_n}{\partial q_i} \Big). \tag{27}$$

Thus, the rate of change of each component of the LRL vector is of order ϵ^n. The components themselves, which are time integrals of the above modulo a constant term, must also be of order ϵ^n. Let the LRL vector initially be of length A_0 and lie along the x-axis, then we have

$$A_x(t) = A_0 + \epsilon^n A_{nx}(t) + O(\epsilon^{n+2}), \tag{28}$$

$$A_y(t) = \epsilon^n A_{ny}(t) + O(\epsilon^{n+2}). \tag{29}$$

Since the square of the LRL vector is related to the energy by

$$\mathbf{A}^2 = 2\mathbf{L}^2 E + 1, \tag{30}$$

the longitudinal deviation coefficient $A_{nx}(t)$ is related to the energy error coefficient by

$$A_{nx}(t) = \frac{1}{\epsilon^n} \frac{\mathbf{L}^2}{A_0} (E(t) - E_0) = -\frac{\mathbf{L}^2}{A_0} H_n(t), \tag{31}$$

which gives no new information. The perpendicular deviation coefficient $A_{ny}(t)$ is best measured in terms of the rotation angle:

$$\theta(t) = \tan^{-1}\Big[\frac{A_y(t)}{A_x(t)}\Big] = \epsilon^n \Big[\frac{A_{ny}(t)}{A_0}\Big] + \cdots \tag{32}$$

To compare algorithms we again extract and compare their rotation error coefficient function $\theta_n(t) = A_{ny}(t)/A_0$ via

$$\theta_n(t) = \lim_{\epsilon \to 0} \frac{1}{\epsilon^n} \theta(t). \tag{33}$$

Since this rotational angle is related to some integral of the energy error function, it is a better measure of the overall error of the algorithm.

V. RESULTS OF COMPARING HIGHER ORDER ALGORITHMS

By use of the triplet construction, we generated 6th, 8th, 10th, and 12th order algorithms from both the Forest-Ruth and Chin's C algorithm. We computed the fractional energy deviation, which is just the negative of the energy error coefficient normalized by the initial energy,

$$\lim_{\epsilon\to 0}\frac{1}{\epsilon^n}\Big[\frac{E(t)}{E_0}-1\Big]=-\frac{H_n(t)}{E_0}. \tag{34}$$

Smaller and smaller time steps ϵ are used until the extracted coefficient function is stablized independent of the time step size. This typically occurs in the neighborhood of $\epsilon = P/5000$, where P is the period of the orbit.

Fig.1 compares the (negative) normalized error coefficient functions for the 4th order Runge-Kutta, Forest-Ruth and Chin's C algorithms over one period of the orbit. The error function for the two symplectic algorithms are substantial only near mid period when the particle is at its closest approach to the attractive center. For symplectic algorithms energy is conserved over one period, or its non-conservation is periodic. Its average energy error is bounded and constant as a function of time. In contrast, the 4th order Runge-Kutta energy error function is an irreversible, step-like function over one period. Each successive period will increase the error by the same amount resulting in a linearly rising, staircase-like error function in time. As noted earlier, the maximum error in Chin's algorithm C is smaller than that of the FR algorithm by a factor of 80. However, this error height comparison at one point is not meaningful. It is better to compare the energy error averaged over one period. This would require the integral of the energy error function. On this basis algorithm C will be better still. While the energy error integral can be done, the same goal can be achieved by monitoring the rotation of the LRL vector.

Fig.2 shows the corresponding error coefficient functions of the rotational angle of the LRL vector. After each period, the algorithms rotate the LRL vector by a definite amount. The error coefficient provides an intrinsic, step-size independent way of comparing this

rotation. In Fig.2 the rotated angle produced by algorithm C is too small to be visible when plotted on the same scale as the other algorithms. The insert gives an enlargement of the details. The rotational angle of the LRL vector appears to be related to some integral of energy error function. Although we have not been able to demonstrate this analytically, numerical integration of the energy error function does give a function similiar in shape to the angle coefficient function, having the same numbers of maxima and minima. For the Runge-Kutta, Forest-Ruth and Chin's C 4th order algorithms, the magnitudes of this rotation cofficient after one period are 2.666, 10.860, and 0.004 respectively. On this basis, algorithm C is better than FR by a factor of ≈ 3000. When the orbit is integrated over many periods the rotational angle from symplectic algorithms increases linearly in a staircase-like manner with time. In contrast, the rotational angle of the Runge-Kutta algorithm shows a quadratic increase over long times, such as a few thousand periods. This result is easy to understand if the rotational angle is related to some integral of the energy error. This quadratic increase in the rotation angle of the LRL vector clearly mirrors the quadratic increase in phase error of the Runge-Kutta algorithm, as discussed by Gladman, Duncan and Candy [4].

Fig.3 and Fig.4 show the results when both the Forest-Ruth and Chin's C algorithms are iterated to 6th order by the triplet product construction. Inserts in both detail algorithm C's intricate structure. As an added comparison we also included results for Yoshida's [8] 6th order algorithm A, which is a product of 7 second order algorithms (8), some with negative time steps. For Yoshida's algorithm, Forest-Ruth and Chin's C algorithm in 6th order, the magnitudes of the rotation cofficients after one period are 11.44, 335.1, and 0.1156 respectively. Yoshida's algorithm is a factor of 30 better than FR, but algorithm C is a factor of 3000 better. Note that if the energy error function is related to the differential of the the angle error function, the zeros of the former would correspond to the extrema of the latter. The four zero crossings of algorithm C's energy error function are clearly reflected in the two maxima and two minima of the corresponding angle error function.

Fig.5 and Fig.6 give results for the 8th order iterated algorithms based on the Forest-

Ruth and Chin's C algorithm. The magnitudes of the angle error coefficients are 1.386×10^4 and 0.4532 respectively, giving a ratio of approximately 3×10^4. Algorithm C retains its characteristic shape in both the energy and the angle error function. Fig.7 and Fig.8 give the corresponding results for the iterated 10th order algorithms. Here the intricate structure in the C algorithm is beginning to be washed out. At this high order, quadruple numeric precision is necessary to extract these coefficient functions smoothly. The magnitudes of the angle error coefficients are now 7.141×10^5 and 17.89 respectively, giving a ratio of 4×10^4. Fig.9 and Fig.10 give similar results for the 12th order algorithms. At this point all structures in the C algorithm are gone. The magnitudes of the angle error coefficients are now 4.473×10^7 and 427.5 respectively, giving a ratio of 1.05×10^5.

The iteration of algorithms A and B of Chin [9] also produced results that are better than FR based algorithms. However, we do not detail their results here because they are at least one or two orders of magnitude inferior to algorithm C.

VI. CONCULSIONS

In this work we have shown that higher order force gradient symplectic algorithms appear to be superior to non-gradient symplectic alogorithms as measured by eneregy conservation and the rotation of the LRL vector. While it has been shown earlier that 4th order force gradient algorithms have smaller energy error coefficients [9], it was not known that this advantage would mulitply dramatically when algorithms are iterated to higher orders. The conclusion that one should draw may not be that force gradient algorithms are better, but that higher order non-gradient algorithms are far from optimal. Secondly, we suggested that the rotation of the LRL vector gives an intergrated measure of an algorithm's merit when tested on the Kepler problem.

The high accuracy of this class of algorithms seemed ideal for long time integration of few-body problems, such as that of the solar system [5]. For such few-body problems, the evaluation of the force gradient is not excessively difficult. It would be useful to examine

the merit of this class of algorithms in more physical applications. The distinct advantage uncovered in this work, that it is better to iterate a 4th order algorithm with all positive time steps, gives further impetus to search for an all positive time step 6th order symplectic algorithm.

ACKNOWLEDGMENTS

This research was funded, in part, by the U. S. National Science Foundation grants PHY-9512428, PHY-9870054 and DMR-9509743.

REFERENCES

[1] *Numerical Hamiltonian Problems* by J. Sanz-Serna and M. Calvo, Chapman & Hall London, 1994.

[2] H. Yoshida, Celest. Mech. **56** (1993) 27.

[3] H. Kinoshita, H. Yoshida, an dH. Nakai, Celest. Mech. **50**(1991)59-71.

[4] B. Gladman, M. Duncan and J. Candy, Celest. Mech. **52**(1991)221.

[5] J. Wisdom and M. Holman, Astrophys. J., **102**(1991)1528.

[6] M. Creutz and A. Gocksch, Phys. Rev. Letts. **63**(1989) 9.

[7] M. Suzuki, Phys. Lett. **A146** (1990)319.

[8] H. Yoshida, Phys. Lett. **A150** (1990)262.

[9] S. A. Chin, Phys. Lett. **A226** (1997) 344.

[10] R. D. Ruth, IEEE Trans. Nucl. Sci, **NS-30**(1983)2669.

[11] E. Forest and R. D. Ruth, Physica D **43**(1990) 105.

[12] M. Campostrini and P. Rossi, Nucl. Phys. **B329**(1990) 753.

[13] J. Candy and W. Rozmus, J. Comp. Phys. **92**(1991) 230.

[14] M. Suzuki, J. Math. Phys. **32** (1991)400.

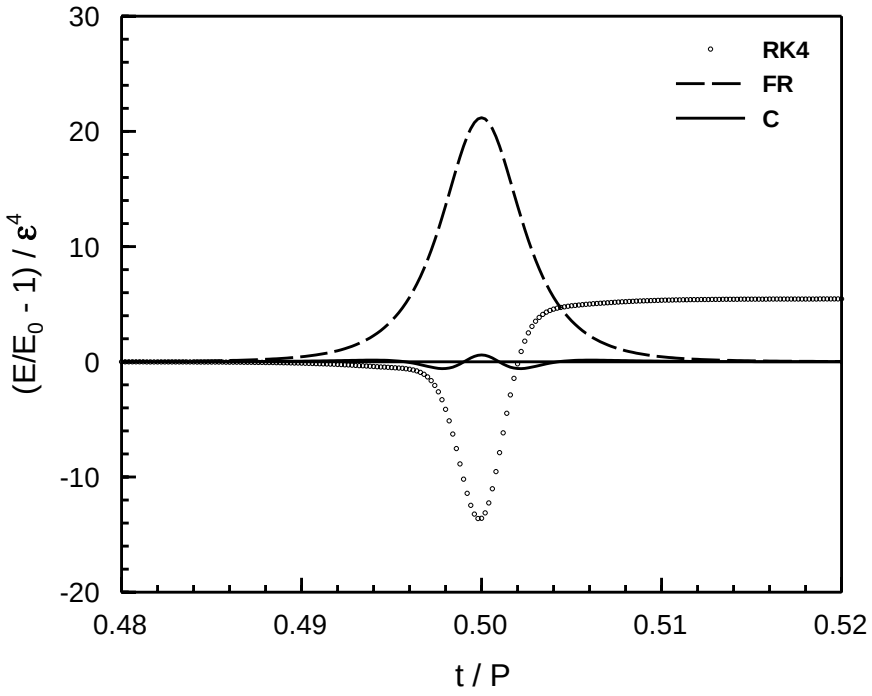

FIG. 1. The normalized energy deviation of a particle in a Keplerian orbit, which measures the step-size independent energy error coefficient $-H_4(\mathbf{p}(t), \mathbf{q}(t))/E_0$. P is the period of the elliptical orbit and ϵ is the time step size. RK4, FR, and C denote results for the 4th order Runge-Kutta, Forest-Ruth, and Chin's C algorithm respectively. The maximum deviations for algorithm FR and C are 21 and 0.27 respectively.

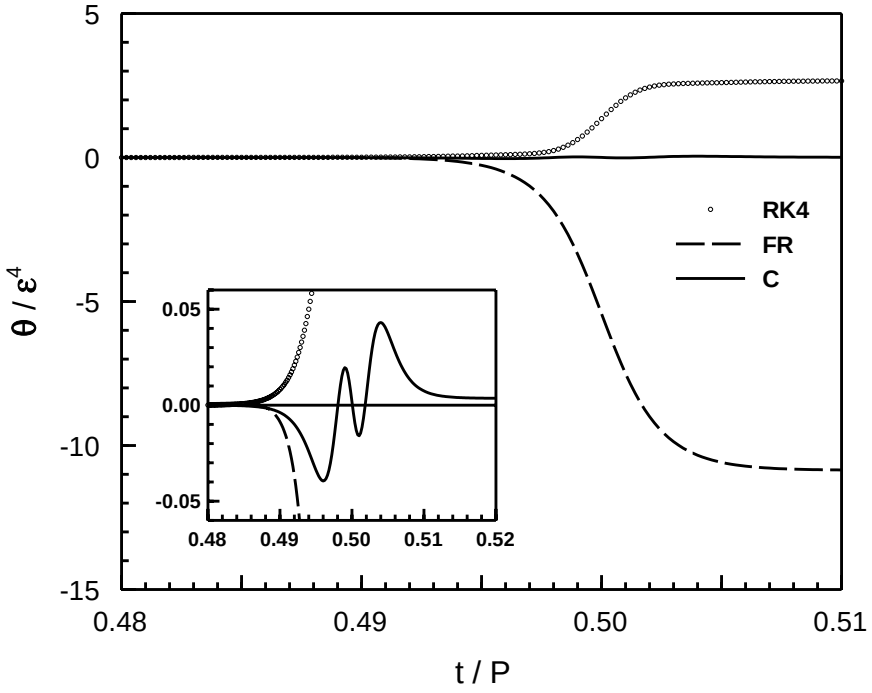

FIG. 2. The step-size independent error coefficient of the rotation angle of the Laplace-Runge-Lenz vector for 4th order algorithms. The LRL vector rotates substantial only when the particle is near mid period, closest to the attractive center. The insert make visible the fine structure produced by algorithm C.

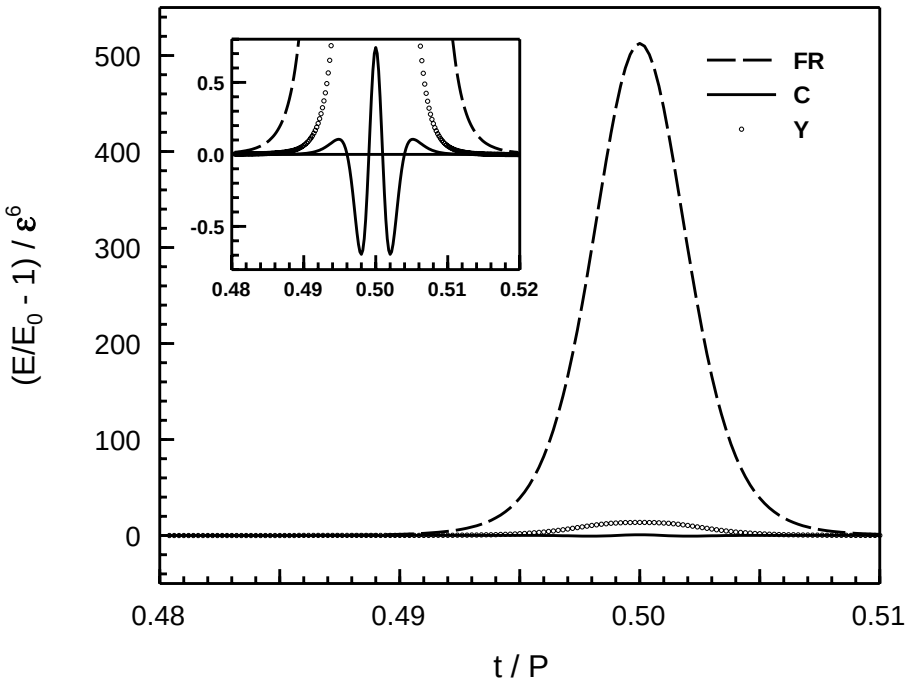

FIG. 3. The normalized energy deviation for 6th order algorithms. RF and C denote 6th order algorithms generated by a triplet product of corresponding 4th algorithms in Fig.1. The insert makes visible the energy devivation of algorithm C, which is not visible in the bigger graph. The maximum deviations for algorithms FR, Y and C are 513, 13.6, and 0.74 respectively.

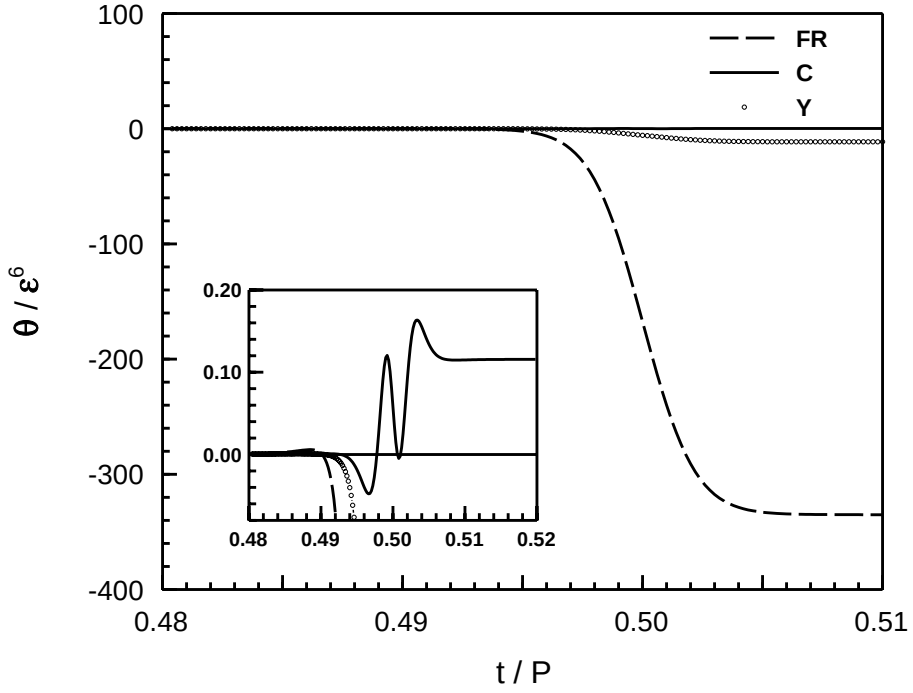

FIG. 4. The step-size independent error coefficient of the rotation angle of the LRL vector for 6th order algorithms as described in Fig.3. The insert makes visible the minute rotation coefficient produced by algorithm C.

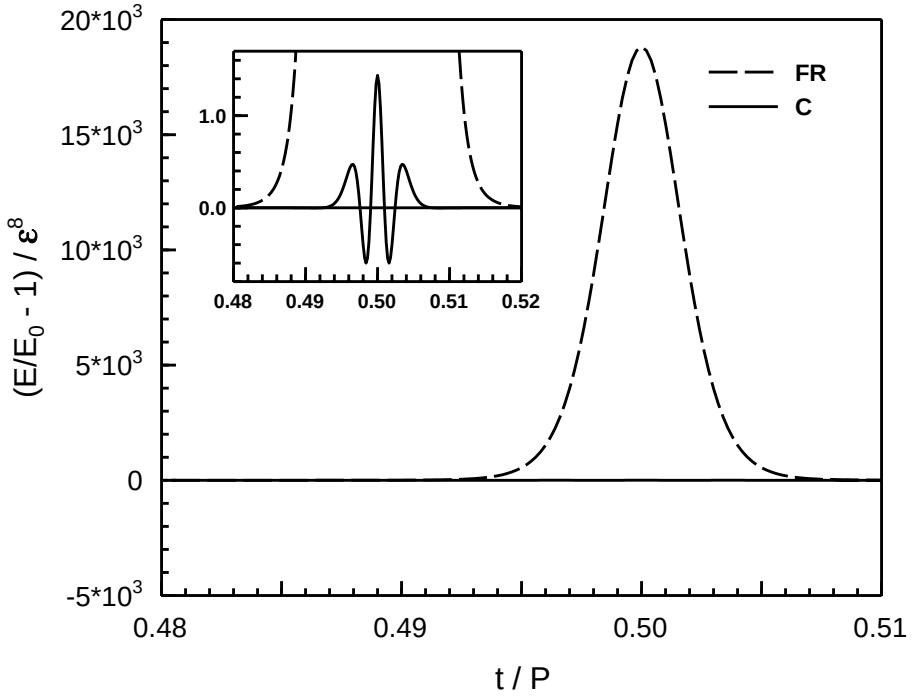

FIG. 5. The normalized energy deviations for 8th order algorithms, as generated by a triplet product of 6th order algorithm described in Fig.3. The insert makes visible the minute energy deviation of algorithm C.

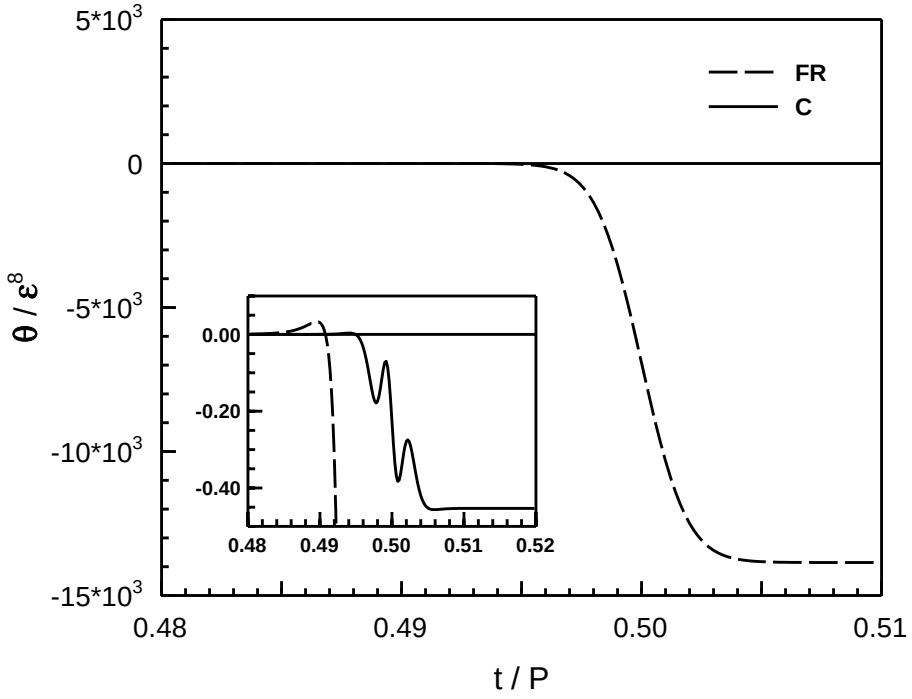

FIG. 6. The step-size independent error coefficient of the rotation angle of the LRL vector for 8th order algorithms as described in Fig.5. In this order, the algorithm C based algorithm begins to rotate in the same sense as the FR base algorithm.

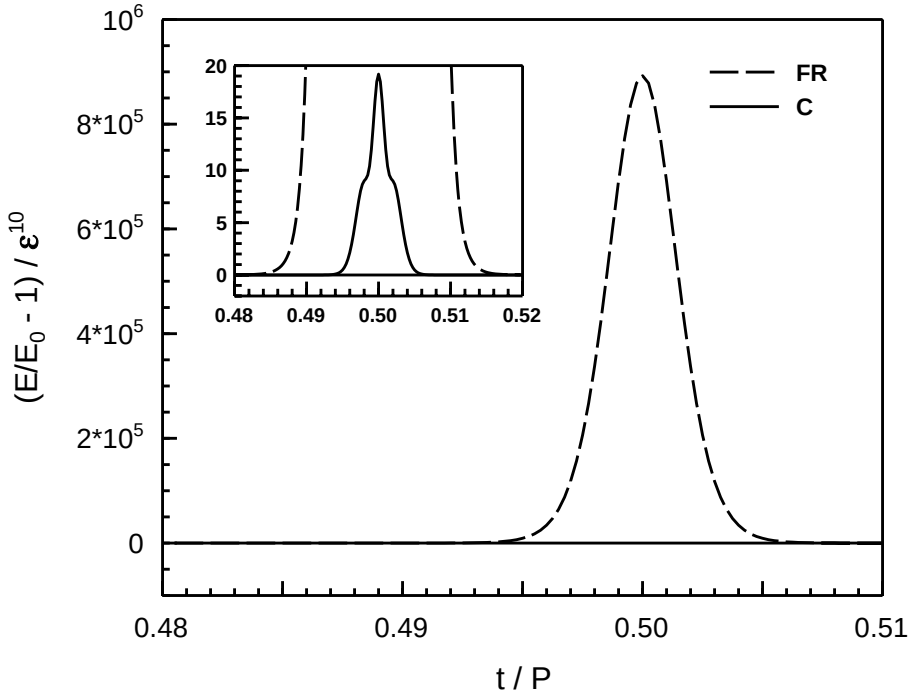

FIG. 7. The normalized energy deviations for 10th order algorithms, as generated by a triplet product of 8th order algorithm described in Fig.5. The insert shows that the characteristic oscillations of algorithm C are beginning to disappear.

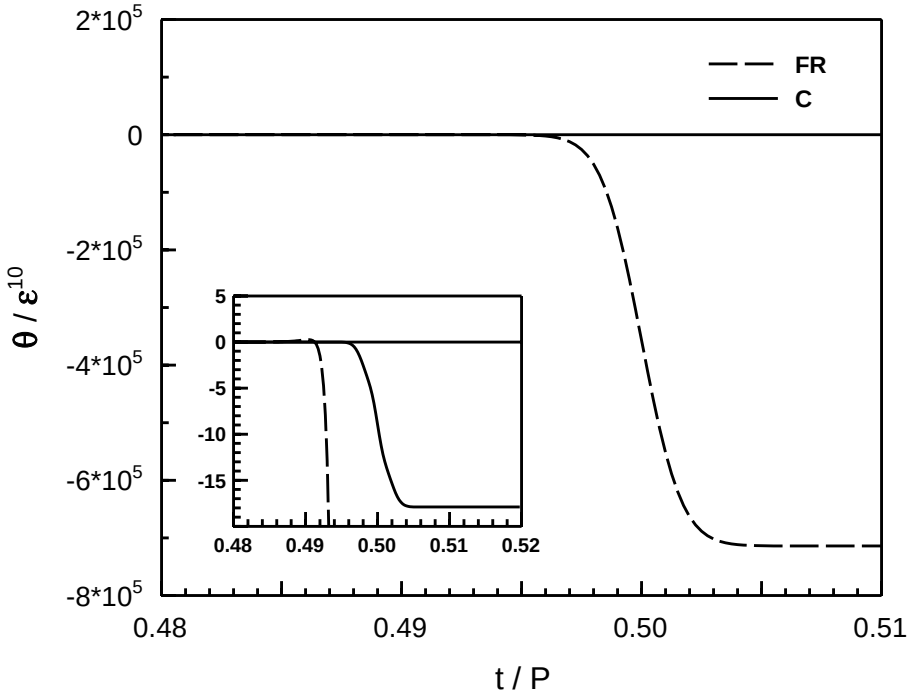

FIG. 8. The step-size independent error coefficient of the rotation angle of the LRL vector for 10th order algorithms as described in Fig.7. The insert shows that the error coefficient of algorithm C begins to look like that of the FR algorithm.

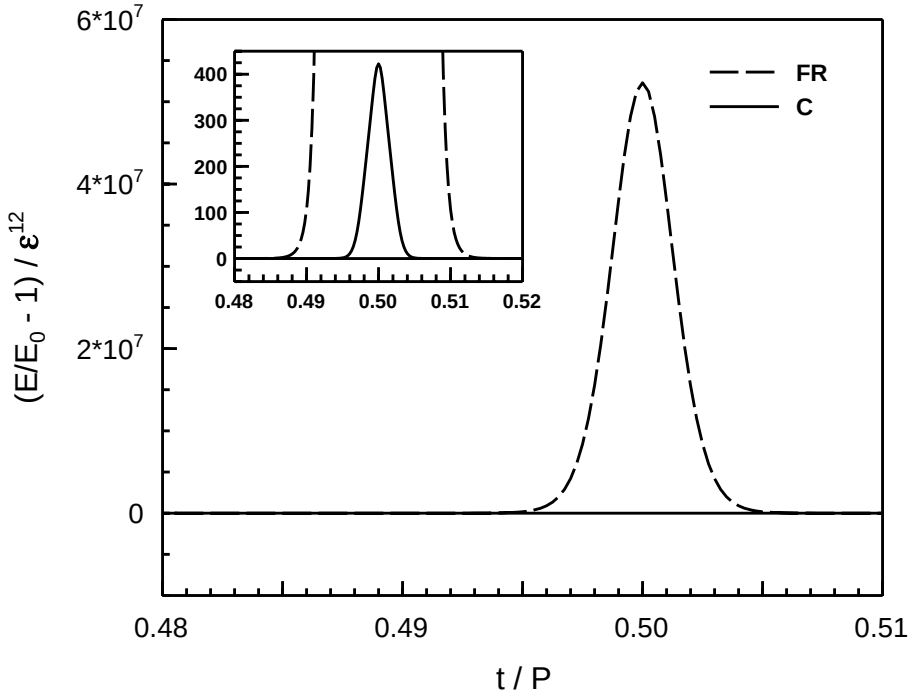

FIG. 9. The normalized energy deviations for 12th order algorithms, as generated by a triplet product of 10th order algorithm described in Fig.7. The insert shows that there is no longer any distinctive structure produced by algorithm C.

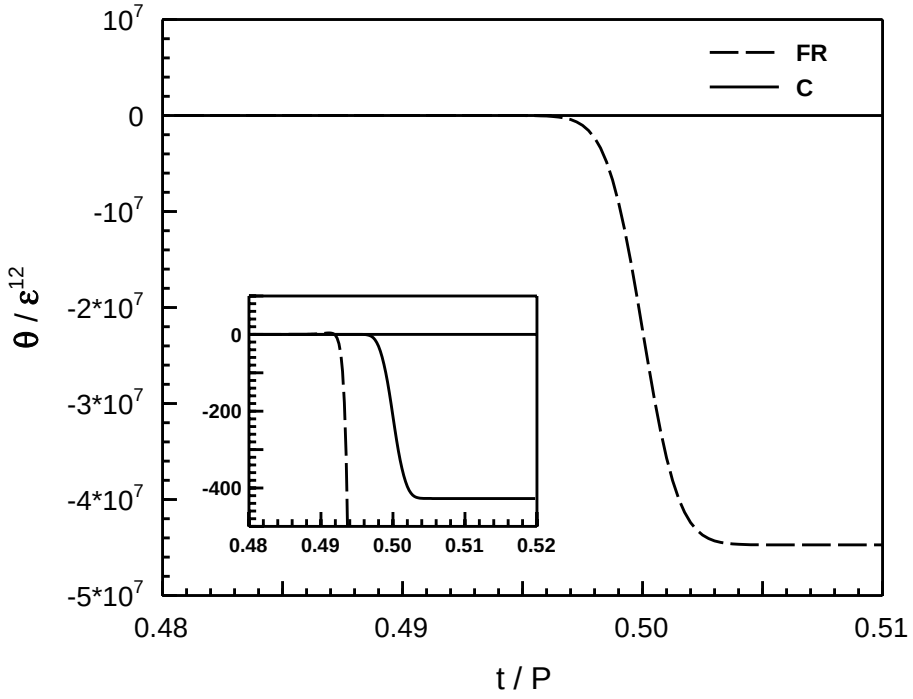

FIG. 10. The step-size independent error coefficient of the rotation angle of the LRL vector for 12th order algorithms as described in Fig.9. The insert shows that both algorithms have converged to a similar step-like error function.